Cincom

SUPRA SERVER PDM

Digest for VMS Systems

P25-9062-50

SUPRA® Server PDM Digest for VMS Systems

Publication Number P25-9062-50

Cincom Systems, Inc.
55 Merchant Street
Cincinnati, Ohio 45246-3732
U.S.A.

PHONE: (513) 612-2300
FAX: (513) 612-2000
WORLD WIDE WEB: http://www.cincom.com

Attention:

Some Cincom products, programs, or services referred to in this publication may not be available in all countries in which Cincom does business. Additionally, some Cincom products, programs, or services may not be available for all operating systems or all product releases. Contact your Cincom representative to be certain the items are available to you.

Release information for this manual

The *SUPRA Server PDM Digest for VMS Systems*, P25-9062-50, is dated January 15, 2002. This document supports Release 2.4 of SUPRA Server with VMS PDM support.

We welcome your comments

We encourage critiques concerning the technical content and organization of this manual. Please take the survey provided with the online documentation at your convenience.

Cincom Technical Support for SUPRA Server PDM

FAX: (513) 612-2000
Attn: SUPRA Server Support

E-mail: helpna@cincom.com

Phone: 1-800-727-3525

Mail: Cincom Systems, Inc.
Attn: SUPRA Server Support
55 Merchant Street
Cincinnati, OH 45246-3732
U.S.A.

Contents

About this book

Using this document

This manual introduces SUPRA Server and associated database access, management, and connectivity tools. It is intended to give a SUPRA Server user a detailed overview of capabilities and features.

Document organization

The information in this manual is organized as follows:

Chapter 1—SUPRA Server overview
Describes the SUPRA Server relational database management system and its tools and architecture.

Chapter 2—SUPRA Server with PDM and RDM support
Describes PDM and RDM support in SUPRA Server for high-volume, update-oriented production processing.

Chapter 3—Using the Physical Data Manager
Describes how to use the PDM, which controls the storage and maintenance of data in SUPRA Server PDM databases.

Chapter 4—Using RDM to access relations
Describes how SUPRA Server uses the Relational Data Manager to provide relational access to physical files.

Chapter 5—Using DBA functions
Describes how to use DBA functions to interact with the Directory. The Directory integrates and controls the other SUPRA Server components. It holds a description of both the logical and physical data structure and how they map to each other.

Chapter 6—Using SPECTRA to retrieve data
Describes SPECTRA, a sophisticated report writer and database access tool. This component is not available for Alpha environments.

Appendix A—SUPRA Server documentation synopses
Provides general information about the SUPRA Server manual series.

Index

Revisions to this manual

The following changes have been made for this release:

- ♦ This manual replaces the former *SUPRA Server Digest*, P26-9065. The contents of that manual have been divided into two separate manuals for SUPRA Server PDM and SUPRA Server SQL.
- ♦ All references to SUPRA Server PDM for VMS Release 2.3 have been changed to 2.4.
- ♦ All references to Windows 95 have been changed to read Windows 95/98.

Conventions

The following table describes the conventions used in this document series:

Convention	Description	Example

`Constant width type`	Represents screen images and segments of code.	`PUT 'customer.dat'`
		`GET 'miller\customer.dat'`
		`PUT '\DEV\RMT0'`

Slashed b (b̸)	Indicates a space (blank). The example indicates that four spaces appear between the keywords.	`BEGNb̸b̸b̸b̸SERIAL`
Brackets [ ]	Indicate optional selection of parameters. (Do not attempt to enter brackets or to stack parameters.) Brackets indicate one of the following situations:	
	A single item enclosed by brackets indicates that the item is optional and can be omitted. The example indicates that you can optionally enter a WHERE clause.	`[WHERE` *`search-condition`*`]`
	Stacked items enclosed by brackets represent optional alternatives, one of which can be selected. The example indicates that you can optionally enter either WAIT or NOWAIT. (WAIT is underlined to signify that it is the default.)	$\begin{bmatrix}\underline{\textbf{(WAIT)}}\\ \textbf{(NOWAIT)}\end{bmatrix}$
Braces { }	Indicate selection of parameters. (Do not attempt to enter braces or to stack parameters.) Braces surrounding stacked items represent alternatives, one of which you must select. The example indicates that you must enter ON or OFF when using the MONITOR statement.	$\textbf{MONITOR}\begin{Bmatrix}\textbf{ON}\\ \textbf{OFF}\end{Bmatrix}$

Convention	Description	Example
Underlining (In syntax)	Indicates the default value supplied when you omit a parameter. The example indicates that if you do not choose a parameter, the system defaults to WAIT.	**(WAIT)** **(NOWAIT)**
	Underlining also indicates an allowable abbreviation or the shortest truncation allowed. The example indicates that you can enter either STAT or STATISTICS.	`STATISTICS`
Ellipsis points...	Indicate that the preceding item can be repeated. The example indicates that you can enter multiple host variables and associated indicator variables.	`INTO :`*`host-variable`*` [:`*`ind-variable`*`],...`
SMALL CAPS	Represent a keystroke. Multiple keystrokes are hyphenated.	ALT-TAB

UPPERCASE
lowercase
	In most operating environments, keywords are not case-sensitive, and they are represented in uppercase. You can enter them in either uppercase or lowercase.	`COPY MY_DATA.SEQ`
		`HOLD_DATA.SEQ`

Italics	Indicate variables you replace with a value, a column name, a file name, and so on. The example indicates that you must substitute the name of a table.	`FROM `*`table-name`*

Punctuation marks	Indicate required syntax that you must code exactly as presented.
	( ) parentheses
	. period
	, comma
	: colon
	' ' single quotation marks
		`(`*`user-id`*`, `*`password`*`, `*`db-name`*`)`
		`INFILE 'Cust.Memo' CONTROL LEN4`

SUPRA Server documentation series

SUPRA Server is the advanced relational database management system for high-volume, update-oriented production processing. A number of tools are available with SUPRA Server including DBA Functions, DBAID, precompilers, SPECTRA, and MANTIS. The following list shows the manuals and tools used to fulfill the data management and retrieval requirements for various tasks. Some of these tools are optional. Therefore, you may not have all the manuals listed. For a brief synopsis of each manual, see “SUPRA Server documentation synopses” on page 77.

Overview

- *SUPRA Server PDM Digest for VMS Systems*, P25-9062

Getting started

- *SUPRA Server PDM VMS Installation Guide*, P25-0147
- *SUPRA Server PDM VMS Tutorial*, T25-2263

General use

- *SUPRA Server PDM Glossary*, P26-0675
- *SUPRA Server PDM Messages and Codes Reference Manual (PDM/RDM Support for UNIX & VMS)*, P25-0022

Database administration tasks

- *SUPRA Server PDM Database Administration Guide (UNIX & VMS)*, P25-2260
- *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130
- *SUPRA Server PDM Utilities Reference Manual (UNIX & VMS)*, P25-6220
- *SUPRA Server PDM Directory Views (VMS)*, P25-1120
- *SUPRA Server PDM Windows Client Support User’s Guide*, P26-7500*
- *SPECTRA Administrator’s Guide*, P26-9220**

Application programming tasks

- *SUPRA Server PDM Programming Guide (UNIX & VMS)*, P25-0240
- *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130
- *SUPRA Server PDM RDM Administration Guide (VMS)*, P25-8220
- *SUPRA Server PDM Windows Client Support User's Guide*, P26-7500*
- *MANTIS Planning Guide*, P25-1315**

Report tasks

- *SPECTRA User's Guide*, P26-9561**

NOTE

Manuals marked with an asterisk (*) are listed twice because you use them for different tasks.

NOTE

Educational material is available from your regional Cincom education department.

1

SUPRA Server overview

SUPRA® Server is an advanced relational database management system based on a three-schema architecture. SUPRA Server provides tools for information retrieval, report generation, application programming, and database management and control.

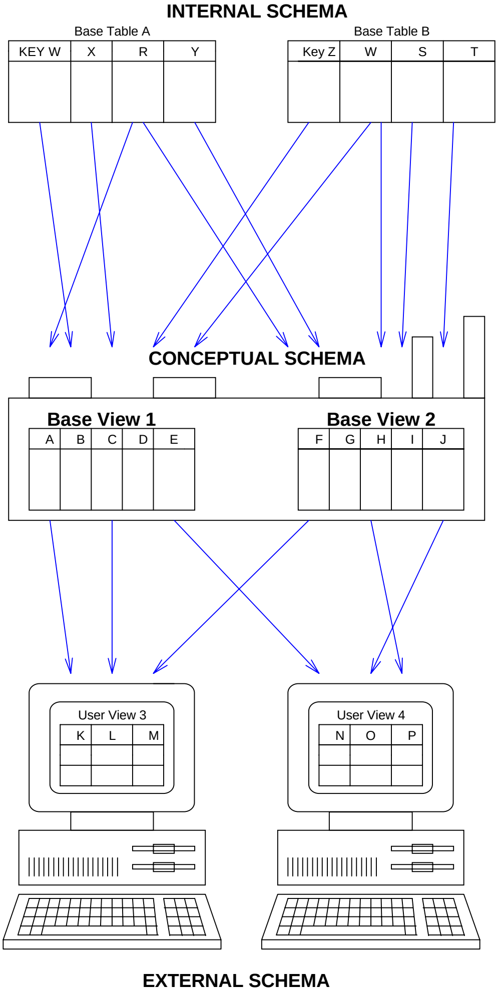
Three-schema architecture

The three-schema architecture implemented by SUPRA Server maintains the table data in three distinct layers:

- **Internal schema.** Describes the structure of physical files and their fields.
- **Conceptual schema.** Defines the base logical views that correspond to the physical files. These view definitions specify the implementation of integrity rules within and between views.
- **External schema.** Describes the logical views whose data is derived from the base views. These derived views describe data for access by applications.

This three-schema architecture means that all data in the database appears to the users as two-dimensional tables. Users can request this data without regard for where or how the data is stored.

The following figure shows the relationships between the three schemas:



The three-schema architecture of SUPRA Server avoids file-redundancy problems and reduces software maintenance costs.

SUPRA Server components

The following figure shows an overview of SUPRA Server with SQL Server, PDM Server, and native file Server support. The key pieces of SUPRA Server include the following:

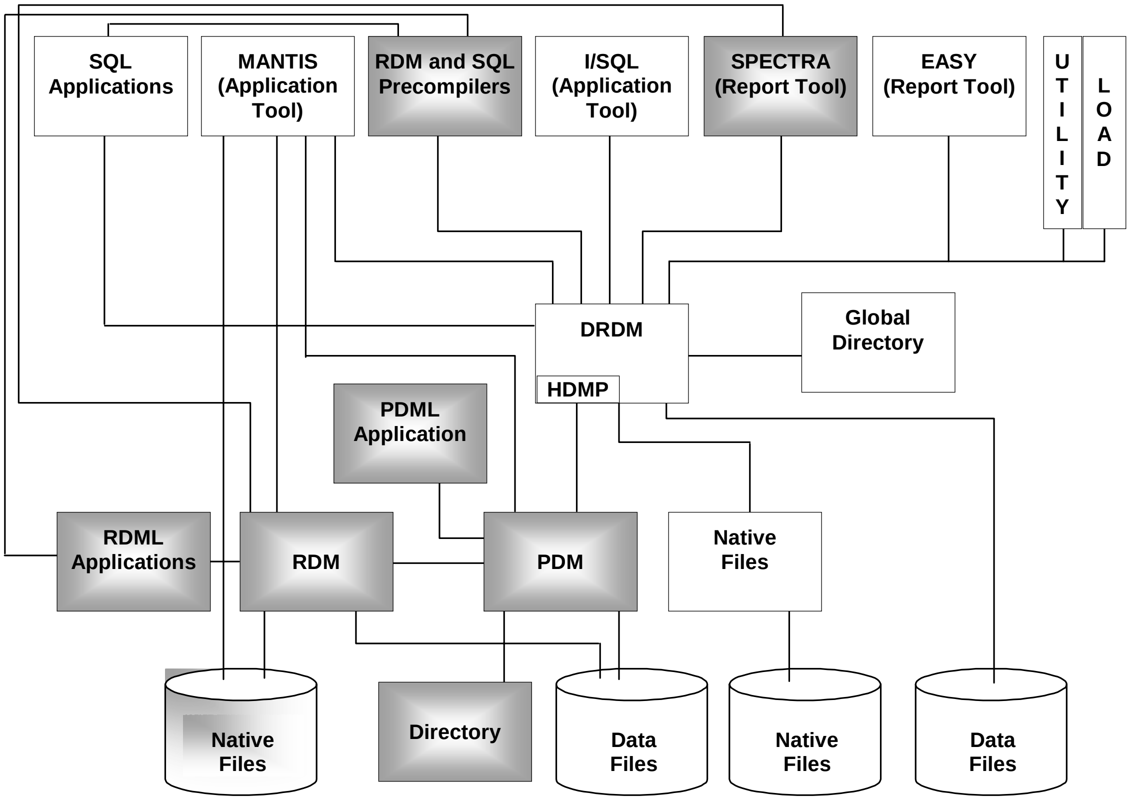
- **Distributed Relational Data Manager (DRDM).** Handles all user access to the database and manages the physical data. The DRDM provides heterogeneous data access to DPDM tables, PDM data, native file data, other DBMS products, multiple file structures, and file management systems.

- **Global Directory.** Controls and maintains the metadata describing the database.

- **Physical Data Manager (PDM).** Manages the data structures of the physical files for databases with a PDM Server. PDM files can be accessed through the Relational Data Manager (RDM) or by applications.

- **Relational Data Manager (RDM).** Processes applications' requests to access PDM data and presents it as though it were arranged in two-dimensional tables.

- **Directory.** Controls and maintains metadata for PDM databases.



SUPRA Server has three options:

- PDM support
- PDM/RDM support
- SQL support

Shaded areas in the above illustration indicate SUPRA PDM components.

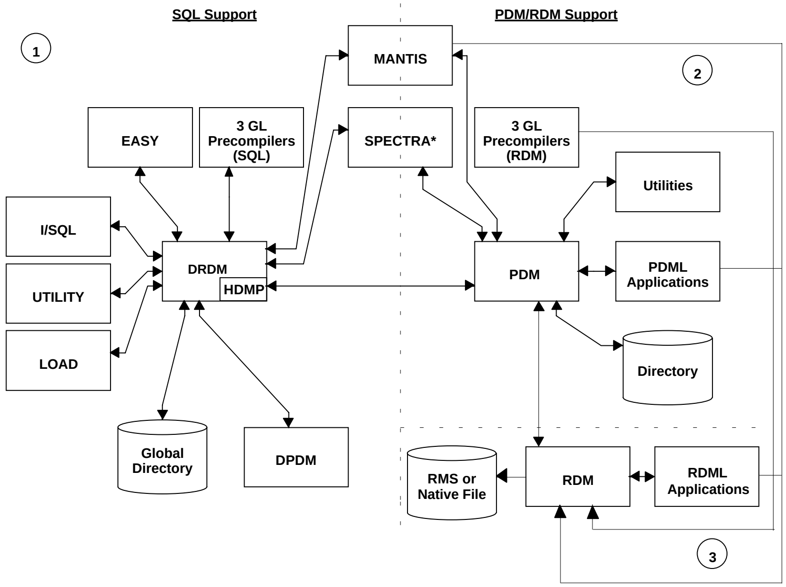
SUPRA Server with PDM support

SUPRA Server with PDM support provides access to data through a Physical Data Manager (PDM), which manages the data structures of the physical files that store data. A Relational Data Manager (RDM) extracts data through the PDM and displays the data in two-dimensional tables.

- **Supported environments.** OS/390, VMS, VSE, UNIX.
- **How the PDM stores data.** The PDM stores data physically in RMS sequential fixed files.
- **How you access data.** Use Physical Data Manipulation Language (PDML) to access data. With RDM support, you use Relational Data Manipulation Language (RDML) commands to the RDM, which extracts data from the PDM and displays the data in two-dimensional tables.
- **Benefits of SUPRA Server with PDM support.** SUPRA Server with PDM support is designed for high-performance, large-scale applications and provides a high degree of control over the storage and maintenance of data.

The following figure presents an overview of SUPRA Server and the major tools available:

- SQL support (group 1)
- SQL support and optional PDM support (groups 1 and 2)
- PDM and optional RDM support (groups 2 and 3)



SUPRA Server with PDM support

The basic element of SUPRA Server with PDM support is the Physical Data Manager (PDM). The PDM maintains data and coordinates database access, processes requests from tools and precompiled programs, and provides backup and recovery tools. The PDM provides a great deal of control over data storage options and is designed for high-volume, online transaction processing.

Components and tools of SUPRA Server PDM

SUPRA Server with PDM support includes the following:

- **PDM (Physical Data Manager)**. A database management system that manages the data structures of the physical files and processes requests for data.
- **RDM (Relational Data Manager)**. An optional database access system that provides relational access to physical files and treats data as though it were arranged in two-dimensional tables.
- **Directory**. A repository of information about the physical data structures, logical data structures, users, and SUPRA Server implementation options.
- **SPECTRA**. (Not available for Alpha environments.) A relational inquiry-reporting tool that provides the ability to retrieve, manipulate, format, and update data.
- **Dynamic indexing**. A tool to allow index access to database files.
- **RMS journaling**. Support for Digital's RMS Journaling, which provides for logging, before image logging, and logical unit of work.

SUPRA Server with PDM support provides a variety of optional tools and functions that let you tailor SUPRA Server to your needs and environment. These tools and functions include the following:

- **DBA utilities**. Tools to organize and maintain data in an efficient, flexible manner. You access most of the utilities through DBA, a menu-driven tool for database maintenance. You can execute some DBA utilities (Fast utilities) from the command level.
- **MANTIS**. An application programming tool.
- **Precompilers**. Allow you to embed RDML statements within application programs. SUPRA Server provides precompilers for COBOL, FORTRAN, and BASIC.

Features of SUPRA Server with PDM support

SUPRA Server with PDM support provides the following features:

- **Multiple file-type support**. The PDM supports the following types of files:
    - Data files (primary and related, including Directory files) that contain user data and Directory information.
    - Index files that are optional and contain pointers to data file records and secondary key definitions.
    - System files that are optional and are for recording statistics, task level recovery, and system level recovery information.
- **Active schema (database) maintainability**. You can perform maintenance on entities in an active schema, or database. Perform active database maintenance by using DBA utilities and DBA functions or by using Batch Directory Maintenance.
- **Storage optimization**. The PDM runs as a multithreaded, detached process. Multithreading allows numerous applications to use the PDM at the same time and leads to efficient use of system resources. Running the PDM as a detached process results in smaller application programs and thereby saves on disk space and memory.
- **Recoverability**. After a failure, you can use task logging and/or system logging to recover the database. Task logging ensures that the PDM saves enough information for all tasks and transactions to perform task level recovery. System logging allows recovery if the task log file is unreadable or unused (on certain environments), or if an updated data file is unreadable and must be reloaded.
- **Data accessibility**. You can access files in four different ways:
    - Directly by primary key.
    - Directly or sequentially by secondary key.
    - Sequentially (forward or backward).
    - For read-only access to secondary key data.

Features of SUPRA Server with RDM support

SUPRA Server with RDM support provides the following features:

- **Testability**. The DBAID utility allows you to test RDM views without affecting the directory to ensure they work correctly before putting them into production use. A subset of the DBAID utility is also available to application programmers to test views used in base programs.

NOTE

CSIDBVER lets you look at PDM data without writing an application.

- **Data accessibility**. You can access files in four different ways:
  - Directly by primary key.
  - Directly or sequentially by secondary key.
  - Sequentially (forward or backward).
  - For read-only access to secondary key data.
- **View validation using domains**. A domain is the set of all permissible values for specified fields. You can use domains to indicate whether a value is valid for a given field and whether two fields are logically compatible.
- **Referential integrity**. Referential integrity ensures that two items representing the same data do not become inconsistent.
- **Tools for optimizing performance**. You can optimize the performance of the RDM by reducing processing time using Global View Support and View Binding.

2

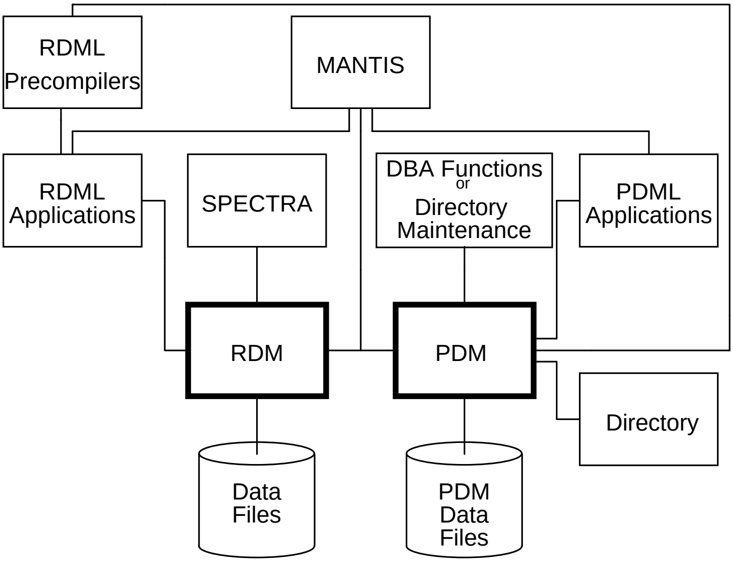
SUPRA Server with PDM and RDM support

SUPRA Server with PDM and RDM support is an advanced relational database management system for high-volume, update-oriented production processing. It contains the following major components:

- **Physical Data Manager (PDM).** Program that processes requests by RDM and other programs to access native SUPRA Server PDM files (see "Using the Physical Data Manager" on page 25).
- **Relational Data Manager (RDM).** Program that processes applications' requests to access data (PDM, VSAM, or RMS) through logical views (see "Using RDM to access relations" on page 35).
- **Directory.** Repository of information about the physical data structures, logical data structures, users, and SUPRA Server implementation options. All requests to the database are handled using information from the Directory (see "Using DBA functions" on page 51).

Overview

The following figure provides an overview of SUPRA Server with PDM support:



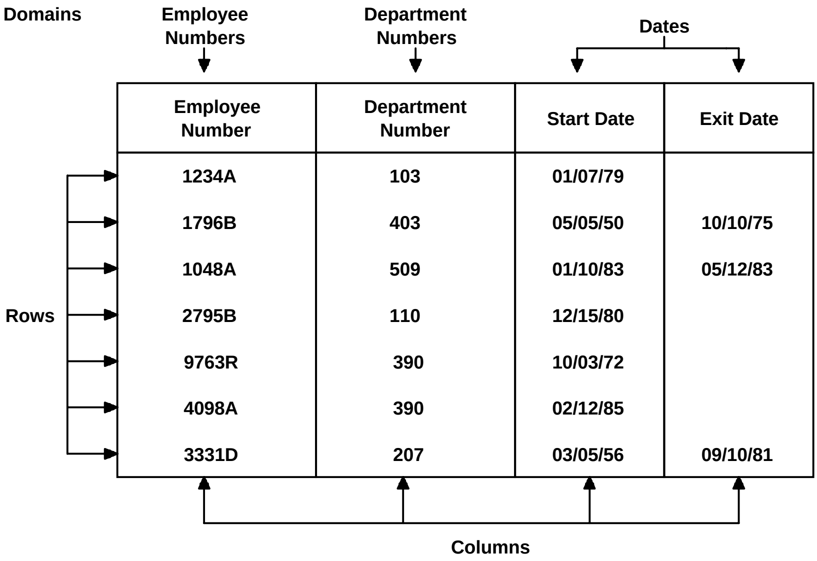
The characteristics and capabilities of SUPRA Server include:

- SUPRA Server supports a relational data structure enabling you to access data through logical views of relations (tables). These logical views of data have a high degree of independence from the needs of any specific application and from the physical structure of the data files.
- SUPRA Server has a three-schema architecture that increases the insulation between your applications' views of data and the physical structure of the data files. Changes to file type or structure involve minimal change to logical views and little or no change to application code.
- SUPRA Server supports several safeguards for the security of your data, protecting it from unauthorized alteration or retrieval.
- SUPRA Server supports access to a variety of file types. It can access its own native PDM files and RMS files.
- SUPRA Server provides for relational and data integrity.
- SUPRA Server provides for recovery of PDM files.

Understanding the relational data structure

SUPRA Server is a relational database. You can view data as if it were a table consisting of rows and columns. In the following figure, Employee Number, Department Number, Start Date, and Exit Date are column names. The combination of one employee number, one department number, one start date, and one exit date make up a row. All rows for all work histories comprise the Work History relation.

A relation has a primary key (a column or combination of columns whose value(s) uniquely identify each row). The primary key of the Work History relation is the Employee Number column. Each column in a relation must have exactly one domain. The domains in the following figure are Employee Numbers, Department Numbers, and Dates. Each value in a column must belong to that column's domain. A domain is a pool of all permissible values for a given meaning. For example, the Department Numbers domain in the following figure consists of all 3-digit numbers 000–999 that might signify departments.



Employee Number	Department Number	Start Date	Exit Date
1234A	103	01/07/79	
1796B	403	05/05/50	10/10/75
1048A	509	01/10/83	05/12/83
2795B	110	12/15/80	
9763R	390	10/03/72	
4098A	390	02/12/85	
3331D	207	03/05/56	09/10/81

If your data is organized and viewed as normalized relations, you receive the full benefit of the relational model. Data normalization examines raw data to determine logical relationships between data items and identifies data dependencies. Normalizing data breaks it into unique occurrences and then restructures it into relations. The objective of normalization is to define relations that are independent of process, free of data redundancies, free of data ambiguities, precisely defined, and independent of file structures.

Ensuring security in SUPRA Server

SUPRA Server with PDM support provides database security at the component and the entity levels. You define all users or groups of users on the Directory. The definition includes a nondisplayed user password. The Directory controls who uses your data and how it can be accessed and manipulated. Only authorized users can access Directory Maintenance, DBA Functions, DBAID, SPECTRA, or RDM. Using the relationship commands on the Directory, you can also restrict access to specific views and relations.

Views can also restrict file/relation access. For example, a PDM file opened with shared-update access can be accessible only to a certain user with READ access on a subset of the columns in the relation.

Using multiple file types

SUPRA Server with PDM support stores data in multiple user-file types. User files can be:

- Native PDM files
- RMS files (VMS RDM only)

The Directory stores detailed, internal-schema information for each file known to SUPRA Server. This information includes:

- File type
- Physical access method
- Data set name and/or ddname
- Block size or control interval size
- Length and displacement of the file’s key (if applicable)
- Length and location of the file’s secondary key(s) (if applicable)
- Descriptions of the physical fields in the file’s records (names, lengths, displacements, types)

When an application requests data through the RDM, RDM refers to the file information from the Directory. Based on this information, RDM calls the appropriate programs to perform the physical accesses to satisfy the request.

3

Using the Physical Data Manager

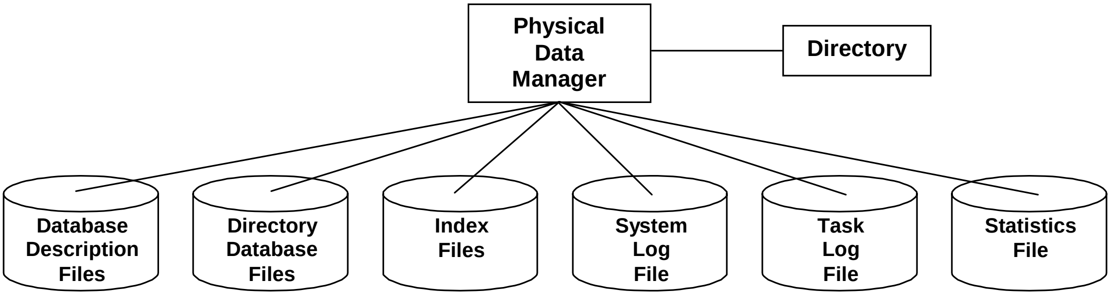
Physical Data Manager overview

The Physical Data Manager (PDM) controls the storage and maintenance of data in SUPRA Server PDM databases. Applications make a request for data to the PDM, which then retrieves the data from the database.

The integration point of this high-performance PDM is the Directory. The Directory stores information about the database and environment managed by the PDM. For more information on the Directory, see “Using DBA functions ” on page 51.

Under VMS, the PDM runs as a multithreaded, detached process. Multithreading allows numerous applications to use the PDM at the same time and leads to the efficient use of system resources. Running the PDM as a detached process results in smaller application programs thereby saving on disk space and memory.

The following figure shows the structure of PDM processing and the types of files handled by the PDM:



The PDM supports a complete range of data structure options. These structures allow you to choose the best technique to fit your needs. You may use any of the following:

- **Hashed**. Provides access to data by hashing of a unique key.
- **Chained**. Allows the chaining of related data records associated with a unique identifier.
- **Indexed**. Provides multiple data-access paths.

Hashing and chaining are the default data structures in the PDM. Given a unique identifier for a primary file data record, the PDM processes the identifier’s value through a hashing algorithm to determine the location of the physical record and then accesses that record. Given the primary file record, the PDM can find any related file records using the same identifier value. Such records are linked together in chains, and the primary file record points to the first and last related record in the chain.

Dynamic indexing

The PDM uses dynamic indexing to speed the retrieval of data through the use of secondary keys. Dynamic indexing can eliminate intermediate application sorting. You use secondary keys to retrieve data in either forward or reverse order. You can define a secondary key for any data field and get the database records back in order by that data field. If two or more data fields make up a secondary key, the data is returned as if the data fields were a sort key. For example, if you define a secondary key using a product field and a date field, the fields are returned in product and date sequence.

The PDM monitors the activity of applications to ensure that they do not perform unauthorized or invalid functions. If an application tries to perform an invalid function, the PDM terminates the function and backs it out, as if it was partially completed. The following are some examples of invalid functions:

- Reading or adding a record with an invalid physical key
- Adding a duplicate primary record
- Requesting a data set or data item that is not in the database
- Using an incorrect data-set type
- Deleting a primary record before all its related data records are deleted

Types of recovery

In addition, the PDM also checks that data sets are opened properly, that they are locked correctly, and that all linkpaths are accurately maintained.

To protect your data from hardware and software failure, the PDM provides task log recovery, system log recovery, and shadow recording.

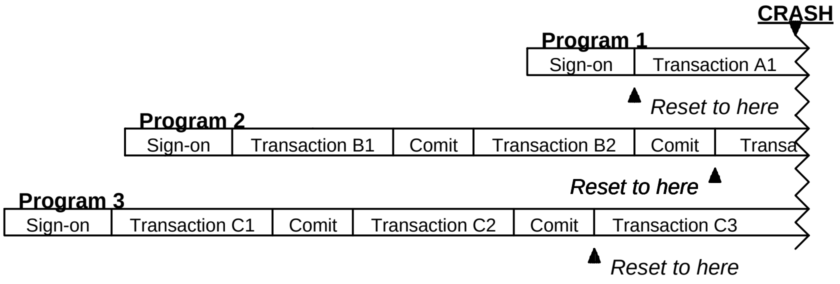
Logging permits recovery, restoration, or resetting of a failed PDM file at a point in the processing cycle where the data is assumed valid. The PDM can record data and function images, and can control information during the processing cycle. Using task logging and task level recovery provides logical integrity and enables restoration of individual tasks and transactions. System logging provides physical database recovery. The PDM also supports RMS Recovery Unit Journaling for RMS data sets.

Task log recovery

Task log recovery records each successful transaction on a task log. You specify task logging when you define each database on the Directory. To make best use of task log recovery, the application programmer divides each program into logical transactions and inserts a COMMIT statement at the end of each transaction. When the program issues a COMMIT statement, PDM performs three actions:

- Writes all updates to the database
- Releases all records held for update
- Resets the task log file for that task

The following figure illustrates three programs executing at the same time. At the end of each logical transaction is a COMMIT statement. When a program fails, the PDM automatically rolls back processing to the last commit point. The only exception is when a program, as in program 1, fails before the first COMMIT statement. Then the PDM rolls the transaction back to the sign-on.

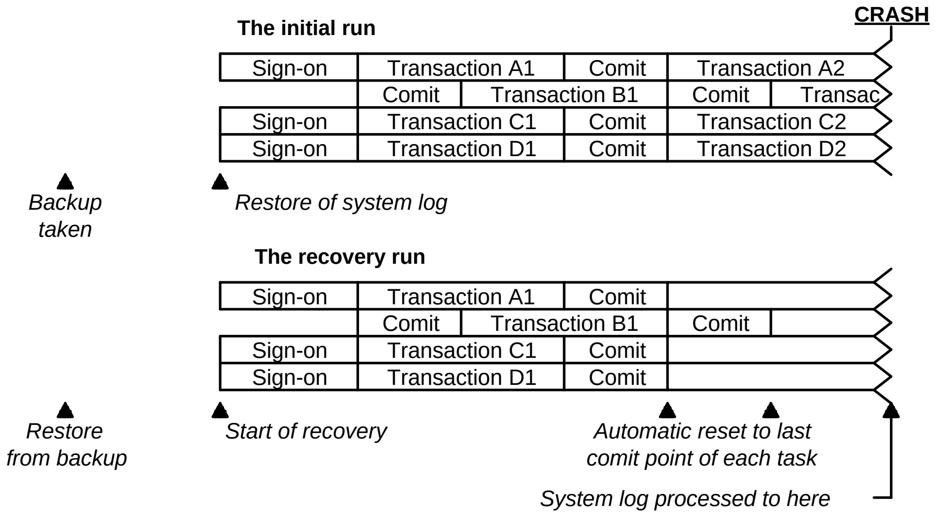


If the system fails, (because of a power failure), the first application to sign on to the database causes the PDM to automatically recover each failed task. In the next figure, the PDM automatically reverses the database updates for Transactions A1, C52, and D3. In other words, the PDM restores the database to its most recent consistent state.

System log recovery

System log recovery logs all completed control functions and images of updated records to a system log. After a failure, system log recovery applies the previously successful updates to a backup copy.

To recover from a system log after a failure, you must first restore the database from a backup copy. Then run system log recovery either online (from DBA command level) or in batch to recover all tasks to the time of failure. Automatic task log recovery then sets all transactions back to their last secure point. The following figure shows the system log applied to a backup database and the subsequent task log recovery.



Shadow recording

Like system logging, shadow recording also allows the user to recover from a disk failure. Every time the PDM writes to a data set in the database, it also writes to the shadow data set thereby duplicating the data set in real time. Then if a disk containing PDM files fails, the PDM switches to the shadow data sets on a different device and continues processing.

Shadow recording saves time because, unlike system logging, the database does not need to be restored from a backup after a disk failure. The automatic switch to shadow data sets by the PDM is quicker than the other recovery methods.

NOTE

For maximum security, use all three types of recovery by shadow recording the task log and system logs.

Using the PDM in a VMS environment

Many aspects of the PDM, such as data structures, data integrity maintenance, and logging, operate the same regardless of the operating system under which they are running; however, some aspects of the PDM, such as the following, are VMS operating-system dependent:

- Initiating the PDM
- Running the PDM in a cluster or network

Initiating the PDM

Because the PDM is a detached process, you must start it before any application can access the database. You can start the PDM in one of the following ways:

- Manually, by executing a PDM initiation command file that runs the PDM start-up program, CSISTR.
- Automatically, at the first attempted database sign-on, using the logical CSI_AutoStart.

Each PDM environment needs a PDM initiation command file, the associated input file, and a PDM start-up resource file. The PDM input file, read by the PDM initiation command file, contains parameters that:

- Control the maximum number of tasks that can concurrently access the PDM.
- Specify ACL checking and/or ULC-based checking for database files.
- Control the maximum number of threads available to the PDM.
- Select whether to redirect PDM messages to a mailbox that is readable from a user-written program.
- Select whether to gather PDM statistics.

The PDM start-up resource file, read by the PDM start-up program, contains parameters that:

- Allocate resources to the PDM process.
- Specify a ULC name for the PDM to start under.

SUPRA Server also provides automatic PDM restart to protect applications and end users against the failure of the PDM machine. Automatic PDM restart is transparent to end users and application tasks. In a single-machine environment, automatic restart occurs when the PDM machine becomes available and may be initiated by the first attempted database access.

In a multimachine environment, automatic PDM restart is controlled by a preferred machine list. This list identifies the machines on which the PDM can run in descending order of preference. If the current PDM machine becomes unavailable, the task restarts the PDM on the highest available machine in the preferred machine list.

Running the PDM in a cluster or network

The PDM runs in both single- and multimachine operating environments. In a single-machine environment, the PDM, its databases, and all accessing tasks run on the same machine. In a multimachine environment, the PDM, its databases, and accessing tasks may all run on different machines.

For all operating environments, SUPRA Server provides the following recovery techniques for data held on SUPRA Server databases:

- Task log recovery to recover data to the last, successful commit point.
- Recovery Unit Journaling to recover RMS data sets to the last successful commit point.
- System log recovery used with task logging to reapply database updates issued after the last commit point.
- Shadow recording to duplicate files on an alternative device.

4

Using RDM to access relations

SUPRA Server with PDM support provides relational access to physical files. The Relational Data Manager (RDM) treats data as though it were arranged in two-dimensional tables (relations) with rows and columns. Logical views define tables for access by RDM.

User applications issue commands in the SUPRA Server Relational Data Manipulation Language (RDML) to RDM. RDM then formulates and issues the appropriate call to the PDM.

With RDM, you can access database information as you view it without concern for its physical location or structure. You can change and restructure your physical database without rewriting or recompiling application programs. RDM handles database navigation, data integrity, data security, and data validation for application programmers and end users.

With RDM, you can access your data files in different ways:

- Directly by primary key
- Sequentially (forward or backward)
- Directly or sequentially by secondary key

Designing and building views

Logical views define relations (tables) consisting of rows (tuples, logical records) and columns (attributes) for access by RDM. Each RDML command must specify a view. In general, a single RDML verb accesses a single row in a relation.

There are two types of views:

- **Base views**. These access physical files. Base views are part of the conceptual schema.
- **Derived views.** These access other views. Derived views are part of the external schema. User views are a subset of derived views.

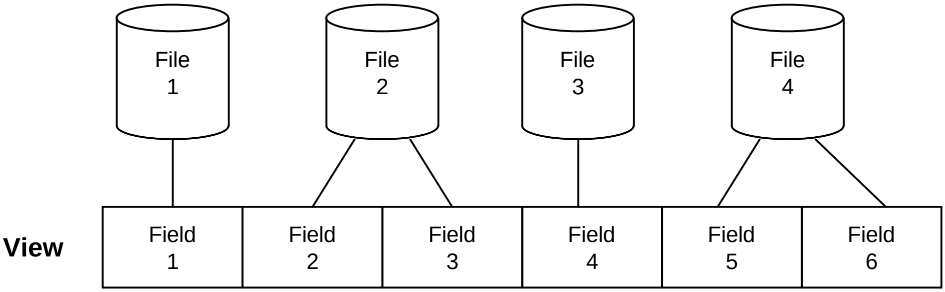
RDM base views can access files of different types, including Cincom's Physical Data Manager (PDM) files, and VMS RMS files. The DBA defines views on the Directory for the data in these files. A single view can access files of different types.

NOTE

A program should not access the same files through both the PDM and the RDM.

Refer to the *SUPRA Server PDM RDM Administration Guide (VMS)*, P25-8220, to explain how to create base views for access to physical files.

A single view can draw fields from several different physical files as shown in the following figure:



A derived view can draw fields from one or several base views. For example, in the following figure, the EMPLOYEE-INFORMATION base view contains 7 fields. If you only need a portion of that data for particular views, you can define user views specifying only the fields you need. The EMPLOYEE-MAIL-ADDRESS derived view uses 4 fields from the base view, and the EMPLOYEE-PAY-RATE derived view uses three. Thus, you retrieve only the data you need for a particular task.

EMPLOYEE-INFORMATION View

Employee Number.	Employee Name	Employee Address	Employee Exemption	Employee Hourly Rate	Employee Social Security Number	Employee Department Number

EMPLOYEE-MAIL-ADDRESS User View

Employee Number	Employee Name	Employee Department Number	Employee Address

EMPLOYEE-PAY-RATE User View

Employee Number	Employee Social Security Number	Employee Hourly Rate

If your applications use derived views rather than base views, you get both the benefit of the three-schema architecture and the insulation from changes to the physical structure.

You can create views with the help of the following software:

- Online DBAID
- Batch DBAID
- DBA Functions

All views are stored on the Directory. You can use Directory Maintenance, DBA Functions, or DBAID to assign your views to specific users or groups of users. You can use the Directory reports to show the definition of a view, who has access to a view, and which programs use the view.

Views can be used with DBAID, MANTIS, SPECTRA, COBOL, PL/I, FORTRAN, and VMS Basic applications. Preopened views and authority to access them can be stored in a Global View file.

Building view definitions with components

A view definition consists of column definitions and access definitions. The column-definition portion of the view precedes the access-definition portion.

When RDM retrieves a row (logical record) using the view, RDM returns one value (or null) for each defined column (except for a constant column, one with the qualifier CONST or UNIQUE CONST). The physical order of the values in the row after retrieval is the same as the order of the column definitions in the view. The chronological order in which the values are retrieved for the row is determined by the access definitions.

You must provide one or more access definitions as part of your view definition. An access definition provides RDM with directions for accessing the desired file(s) or view(s). (Base views access files, derived views access views.)

The following is an example of a base-view definition. Each line except the last defines a column. The last line is the access definition.

```
View name:  BRANCH
KEY  BRANCH-NUMBER
      BRANCH-NAME
      BRANCH-ADDRESS
      BRANCH-CITY
      BRANCH-STATE
      BRANCH-ZIP-CODE
REQ  BRANCH-REGION
      BRANCH-DELIVERY-ROUTE
      BRANCH-SALE-ROUTE
      BRANCH-STAFF-QUOTA
ACCESS BRANCH WHERE BRANCH-NUMBER = BRANCH-NUMBER ALLOW ALL
```

Validating view columns with domains

A domain is the set of all permissible values for a column. Domains are important because they let the business rules be defined and enforced without program maintenance. To create domains on the Directory, a DBA can use DBA Functions or Batch Directory Maintenance. The DBA creates domains using a list of permitted values or by specifying a combination of data-field size, data type, and data-value range. Each domain must be identified by a unique name.

Domains provide two types of validation:

- They indicate whether a value is valid for a given field
- They indicate whether fields are logically compatible

For example, if a customer number is a 5-digit decimal integer, its field should be assigned to a domain defined as the set of all 5-digit decimal integers. RDM can then reject values of the wrong length and type. If an employee number is a 5-digit decimal integer, its field should also be assigned to a domain defined as the set of all 5-digit decimal integers; however, these two fields should not be assigned to the same domain, because they are not logically comparable. It makes no sense to search for an employee with the same number as a customer.

When RDM processes a request to add or update a row, it verifies that the new value is valid by checking the domain for the column. If the value is invalid, RDM returns an error status to the application.

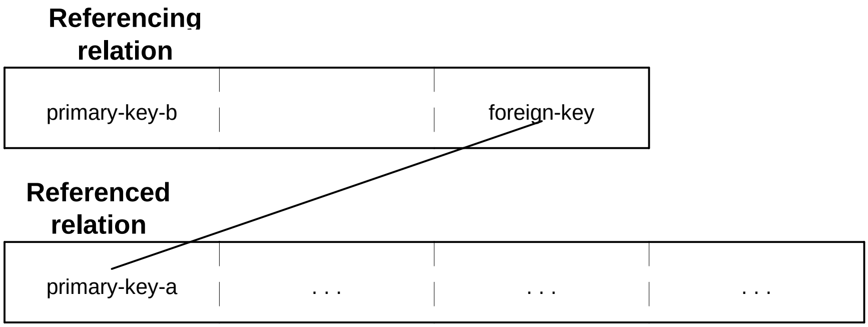
When a view definition indicates that a column in one relation represents the same data as a column in another relation (the foreign key in a referencing relation and the primary key in the referenced relation), RDM verifies that the fields for those columns belong to the same domain.

Adding integrity constraints to base views

If a combination of column(s) in one relation represents the same data as the primary key in a second relation, then:

- The column(s) in the first relation comprise a foreign key.
- The first relation is a referencing, or source, relation.
- The second relation is a referenced, or target, relation.

The following figure shows how one relation references another with a foreign key:



You must identify any foreign keys in your base views so that RDM can maintain referential integrity. Referential integrity ensures that two items representing the same data do not become inconsistent.

RDM supports the following referential integrity rules:

- A foreign key value must exist in the referenced relation as a primary key. In other words, a primary key value must exist for each foreign key value in a referencing relation.
- Null values can be allowed for a foreign key.

RDM checks for referential integrity in two ways:

- **Foreign key value integrity.** When inserting or updating a row that contains a foreign key, the foreign key value must either point to a valid primary key in the referenced relation or be null. Otherwise, RDM will not perform the insert or update.
- **Deletion integrity.** RDM will not delete a row in a referenced relation if the row's primary key is referenced by one or more foreign keys. All foreign keys with that primary key's value must be deleted or set to null before RDM can delete the referenced row.

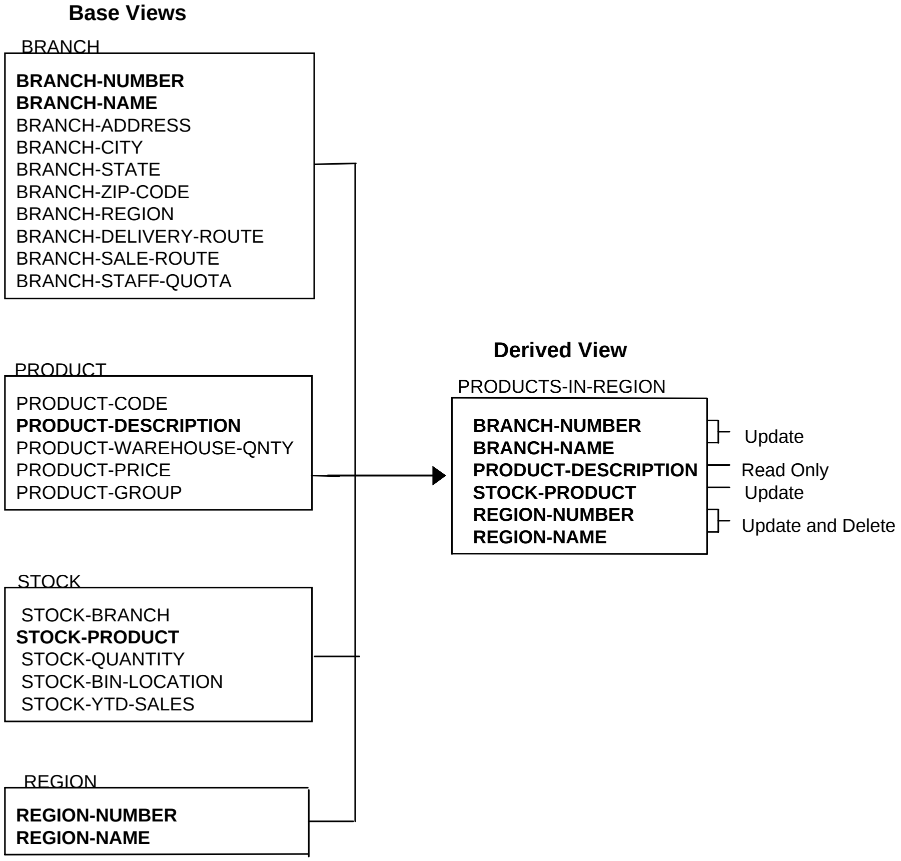
Creating derived view definitions

Derived views access other views instead of accessing physical files. An ACCESS clause specifies a view name instead of a physical file name.

For example, the following derived view PRODUCTS-IN-REGION draws data from the base views BRANCH, PRODUCT, STOCK, and REGION to list all the products in stock in a region.

```
View name:  PRODUCTS-IN-REGION
KEY  REGION-NUMBER
      REGION-NAME
KEY  BRANCH-NUMBER
      BRANCH-NAME
      STOCK-PRODUCT
      PRODUCT-DESCRIPTION
ACCESS REGION WHERE REGION-NUMBER = REGION-NUMBER ALLOW UPDATE
  DELETE
ACCESS BRANCH WHERE BRANCH-REGION = REGION-NUMBER ALLOW ALL
ACCESS STOCK WHERE STOCK-BRANCH = BRANCH-NUMBER ALLOW ALL
ACCESS PRODUCT WHERE PRODUCT-CODE = STOCK-PRODUCT
      AND STOCK-PRODUCT = STOCK-PRODUCT
```

The following figure illustrates the base views, the derived view, and the Update options specified for the derived view:



A derived view can have more restrictive access than its source views, but not less restrictive access. For example, if the base view allows only read access, the derived view cannot allow updates to a field from the base view, even if the derived view specifies ALLOW UPDATE.

When defining a derived view, you need not enter the access definitions for the integrity constraint. You need not rewrite a derived view if the physical file for the BRANCH relation is broken apart into different files or put into another file with a different name.

Accessing data with views

Application programmers access a view through program logic. With RDM, these programs are insulated from the physical environment, including the Physical Data Manager (PDM), data structures, and access and navigation strategies. A programmer uses the Relational Data Manipulation Language (RDML) precompiler to compile source statements into executable code. RDM refers to information from the Directory to retrieve the information required for the view and to access files of any supported type. Supported file types include native SUPRA Server Physical Data Manager (PDM) files and VMS RMS files.

The programmer can write validation user exits to access other types of files. Programmers can write applications using VMS Basic, FORTRAN, COBOL, PL/I, or MANTIS and can access views on the Directory using simple RDML instructions within their programs. Programmers can manipulate data in a program using the four RDML commands: GET, INSERT, UPDATE, and DELETE.

End users can use views online through SPECTRA, a query facility with update capabilities. See “Using SPECTRA to retrieve data” on page 61.

Optimizing performance

You can optimize the performance of RDM in several ways:

- **View binding.** This allows you to store an open version of a view on the Directory. Binding reduces processing time for view requests that cause a view to be opened.
- **Global View support.** This allows you to have certain views open when the database starts. This saves RDM the processing overhead of opening views when the application program first accesses them.
- **Storing the RDM program in shared memory.** (Installed as a known image under VMS.) This allows all applications to share the same copy of the RDM program. This saves application memory space and tends to reduce the paging rate. This is the default for VMS.

Maintaining current programs

RDM uses several checks to ensure that your program is current and that any views it uses are the same as other separately compiled modules in the application program. When an application program issues an RDML command, RDM checks to see if the view, as defined in the Directory, is still correct for this program's use. The checking includes column existence, column length, type and number of decimal places. If the view is not correct, RDM returns an error status; then the program must be recompiled.

Application systems are often composed of several separately compiled programs which depend on a common definition of data terms. These programs call each other to perform special tasks. RDM checks on each RDML call to ensure that the definition of the view is the same for each program. If a program or subroutine is compiled with the same view as another program or subroutine and the view definition does not match, RDM generates an error message. The column (attribute) list generated by the RDML precompiler at precompile time contains the data used to perform this error checking.

Insulating programs from change

RDM insulates application programs from changes to the physical database. For example, moving a column (attribute) from one relation to another has no impact on application programs. In most cases, you can change a view design without affecting the application programs. For example, adding a new column to a view does not affect a program.

Changes to relations include changing a relation type. Usually, RDM insulates application programs from these types of changes. However, the view must be modified and rebound, if previously bound, and reglobalized, if previously globalized, or both.

Physical changes (changing the characteristics of a column in a view) may require changes to the program logic and recompilation. Not every program using a view needs to be recompiled; only recompile the programs that use the column in their user view. The view must also be modified and rebound.

Adding or deleting indexes, secondary keys, or linkpaths may change the behavior of unbound, unglobalized views. RDM selects the access strategy at view open time, and the addition of secondary keys or indexes may alter this selection. If you want a bound or global view to take advantage of a newly defined index or secondary key, you must first rebind the bound view or reglobalize global views or do both.

Creating and testing views with DBAID

It is important to define and test your views to ensure they work correctly before putting them into production use. The DBAID utility allows you to do the following:

- Define a new view without affecting the Directory.
- Open the view.
- Issue RDML statements.
- Examine the results.

You can then change the view if necessary, reorder the view for efficiency, or try different navigation methods until the view works the way you want it to.

If the Directory is available for update, you can instruct DBAID to save your view onto the Directory with the SAVE command. As soon as the SAVE is processed, the view is available to application programs unless a bound version of the view exists. You can bind a view using the DBAID command BIND. You can also take existing views from the Directory, change them for new requirements, and test them to verify that they still work, without affecting the view stored on the Directory or the Global View file.

A subset of the DBA commands is available to application programmers. This subset allows application programmers to use the DBAID utility when constructing programs using views. Programmers can use the DBAID utility to see how the views work and to determine how to design the application based on this information.

Access to the DBAID commands is controlled by the name used for sign-on. If the user is the DBA, as defined on the Directory, then all of the commands are recognized by DBAID. However, if the signed-on user is not the DBA, only certain commands are available. The application programmer cannot define new views or edit existing views. The programmer can access only those views related to the user ID in the Directory file or the Global View file and can access only the data available through those views.

You can use DBAID in a batch or online environment. DBAID provides the following types of commands:

- **System.** Display information about the currently executing DBAID utility. These include commands to display current users, to display active views, and to save a view created through DBAID.
- **Editing.** Change existing view definitions and create new views for testing.
- **RDML.** Test a defined view against the database.
- **View Display.** Display the fields and descriptions for an open view.
- **Statistics.** Gather and print statistics.

Obtaining RDM reports and statistics

For information on obtaining reports and statistics, please refer to *SUPRA Server PDM Directory Views (VMS)*, P25-1120, and to the section titled “Generating RDM reports” in the *SUPRA Server PDM RDM Administration Guide (VMS)*, P25-8220.

5

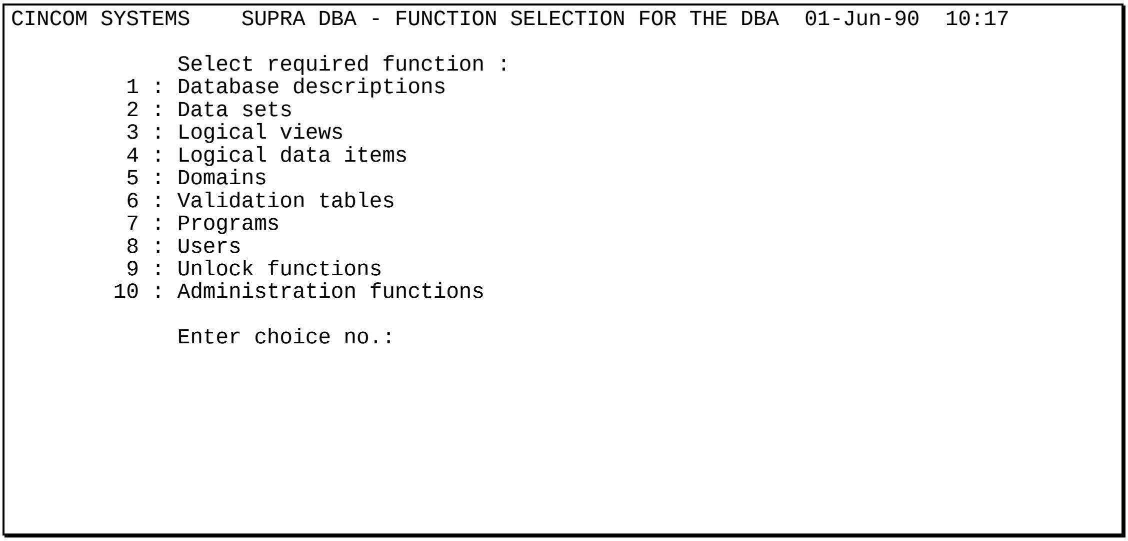
Using DBA functions

The Directory integrates and controls the other SUPRA Server components. The Directory holds a description of both the logical and the physical data structures and how they map to each other. You use DBA functions to interact with the Directory.

Selecting DBA functions

DBA consists of a hierarchy of screens with functions to define and maintain both the physical data structure (the database description) and the logical data structure (views). In addition, DBA offers a number of administrative functions such as defining users and running recovery.

The following figure shows the DBA Function Selection menu:



```
CINCOM SYSTEMS    SUPRA DBA - FUNCTION SELECTION FOR THE DBA  01-Jun-90  10:17

             Select required function :
         1 : Database descriptions
         2 : Data sets
         3 : Logical views
         4 : Logical data items
         5 : Domains
         6 : Validation tables
         7 : Programs
         8 : Users
         9 : Unlock functions
        10 : Administration functions

             Enter choice no.:
```

The following table describes each function:

Function	Activity
1. Database descriptions	Create, examine, modify, copy, and delete database descriptions, including associated buffer allocation, data sets, logging files, and so on.
2. Data sets	Create, examine, modify, and delete data sets independently of the database description.
3. Logical views	Initiate an EDIT/EDT interface to create, examine, modify, and delete base and derived views. Open, save and bind views on the Directory. Define user-to-view access authority.
4. Logical data items	Create, examine, modify, and delete logical data items associated with existing physical data items. Connect logical data item to domain.
5. Domains	Create, examine, modify, and delete domains. Connect domain to logical data item. Connect domain to validation table.
6. Validation tables	Create, examine, modify, and delete validation tables.
7. Programs	Examine and delete programs from the Directory. List which views a program uses.
8. Users	Define users, passwords, and access authority levels.
9. Unlock functions	Reset status of database descriptions, data set descriptions, views, logical data items, and programs after an abnormal system close.
10. Administration functions	Access the administrative functions to format physical files, recover from a system log, run the DBA utilities, expand and reset related data sets, access the index format and populate and check functions.

The SUPRA DBA functions described in the preceding table fall into the following categories:

- **Database description.** Create and maintain the physical data structure. (Function 1, Function 2, Function 5, and Function 6.)
- **View.** Create and maintain the logical data structure   (Function 3 and Function 4.)
- **Unlock.** Reset Directory entities after an abnormal system close. (Function 9.)
- **Administrative.** Control Directory usage, create and modify the structure of physical database files without losing user-entered data and recover after a system or program failure. (Function 7, Function 8, and Function 10.)

Creating and maintaining a database description

You use the database description functions to define the physical structure of a database to the PDM. A SUPRA Server database may consist of many data sets. A data set contains one or more records made up of one or more data items. Each data item may be connected to a domain defining validation criteria including default value, null value, validation table, and/or user exit. (See the representation of the physical structure of data in the Function Selection menu under "Selecting DBA functions" on page 52.) A database description consists of the database details and a definition of each data set connected to that database.

You can create data sets that are connected to one or more databases, or not connected to any database. A data set definition consists of the data set definition and the record layout.

After defining a database, you must validate and compile it to create a compiled database description file (a physical file with a default name of database-name. MOD or .mod) that holds a description of where the data is physically stored. The PDM refers to this file to locate the physical data.

Before SPECTRA, MANTIS, and application programs can use the valid, compiled database to store data you must do the following:

1. Use the Format function to create and format the physical files that will hold data.

2. Use the Logical Data Items function to define logical names for the physical data items.

3. Use DBAID or the DBA Logical View function to design, write, and test logical views.

Function 11, Administration functions, displays the Format function to create and format the physical files assigned to hold the data sets and the task and system logs. The Format function initializes all files and establishes all the control records needed by the PDM.

Creating and maintaining views

After creating a valid, compiled database, formatting the physical files, and defining logical data item names, you use the Logical View functions to create views of the physical database.

The Logical View functions initiate a VMS EDIT/EDT interface for maintaining views. This interface offers the full screen-based, editing techniques to enter and update the text of a view.

You create two types of views:

- **Base views**. These access logical data items directly.
- **Derived views.** These access only columns from other views.

SPECTRA, MANTIS and application programs may access both types of views to retrieve data from the database, although it is better to use derived views where they exist.

Controlling directory usage

RDM automatically enrolls programs onto the Directory after successful preprocessing. Option 7, Programs, from the DBA Function Selection menu allows you to examine the details about a program, modify the comments and list the views the program uses. You can also use this option to delete programs from the Directory, which ensures absolute control over Directory and view usage.

Users cannot sign on to the Directory until you define user identities for them. Option 8, Users, from the DBA Function Selection menu defines user identities and authorization on the Directory. You control user access to the Directory by specifying one of the following user authorities for each user:

User authority	Description
PRIVILEGED (PRIV)	Access to all users and functions.
DATABASE ADMINISTRATOR (DA)	Access to all users and functions except Privileged.
DEVELOPMENT PERSONNEL (DP)	Access to database, data set, read-only access to RDM, and program functions.
READ ONLY (READ)	Read-only access to databases and data sets.
RDM USER (RDM)	No access to DBA functions. Used by RDM programs.

Changing the structure of an active database

You may need to change the structure of a database after it has been released for use. For instance, you may need to change the length of a data item to conform to new standards, or you may need to increase the maximum number of records a data set can hold. The DBA utilities and the Expand and Reset functions from the Administration Functions menu modify the structure of a database without losing previously stored data.

The DBA utilities unload user data from the database so that you can change the physical characteristics and record layout of the data files. Valid modifications include increasing the length of data items and adding or removing data items and linkpaths. DBA utilities unload the data from the original data files into a work file; then they reload the data from the work file into the new data files with the modified layouts. The DBA utilities also produce performance statistics for specified databases and can run both interactively and in batch mode.

The Expand function enlarges a related data set. This function also automatically updates the description of the data set and the database with new file allocation values.

The Reset function changes the control interval load limit of a related data set. After altering the limit, this function automatically updates each control record in a file to reflect the new limit. It also updates the specifications in the data set and database descriptions.

Along with DBA utilities and the Expand and Reset functions invoked from DBA, SUPRA Server offers a set of Fast utilities to change the physical file characteristics of a database. Fast utilities use a command line interface operating both online and in batch mode. The command line interface changes one data set per run. To change multiple data sets in a single Fast utilities run, you create and submit a text file containing all modifications. Refer to the *SUPRA Server PDM Utilities Reference Manual (UNIX & VMS),* P25-6220, for a description of both the DBA utilities and Fast utilities.

Transferring a database from one directory to another

The Batch Directory Maintenance utility (DIRM) allows you to unload the database definition, logical views, and user-to-view permissions from one Directory and load them onto another Directory. The transferred definition includes recovery logs, file specifications, base and derived views, domains, and validation tables, but excludes user-entered data. DIRM is an alternative to the DBA utility for the VMS environment.

DIRM uses a batch script file for exporting and importing metadata, but the utility itself is interactive and menu-driven. Using DIRM, you can unload the metadata for an existing database description or data set from the Directory into a text file containing Batch DIRM input statements. You can then apply your dictionary updates online using the DBA utility, followed by an unload to have a Batch DIRM copy of your updates for later use.

A Batch DIRM statement consists of a database entity category (BUFF for buffer), followed by a command stating how that entity will be affected (ADD), followed by one or more parameters describing which part(s) of the entity the command will act upon (NAME, TYPE). The categories that can be acted upon with Batch DIRM statements include the following:

- Buffers
- Comments for the previous object
- Database descriptions
- Physical data items
- Domains
- Data sets
- Physical file attributes
- Indexes
- Logical data items
- Record codes
- RMS keys
- Secondary keys
- System logs
- Task logs
- SUPRA Server user names
- Validation tables

For details on the Batch Directory Maintenance utility, refer to the *SUPRA Server PDM Utilities Reference Manual (UNIX & VMS)*, P25-6220.

Recovering databases

The Recovery function restores a database after a system failure or a permanent input/output error on a data set. This function gives access to system level recovery; however, system log recovery works only if you define a system log during database definition. See "Using the Physical Data Manager" on page 25 for more information on recovery.

6

Using SPECTRA to retrieve data

SPECTRA is a sophisticated report-writer and database-access-retrieval tool. It is available as a separate component of SUPRA Server with PDM support.

Overview

NOTE

SPECTRA is not available for Alpha environments.

With SPECTRA, you can use simple English commands to select and produce reports on the data contained in SUPRA Server tables and views. You can create complex reports including customized titles, statistics, totals, and grand totals. SPECTRA is a fully interactive system with online documentation and Help facilities.

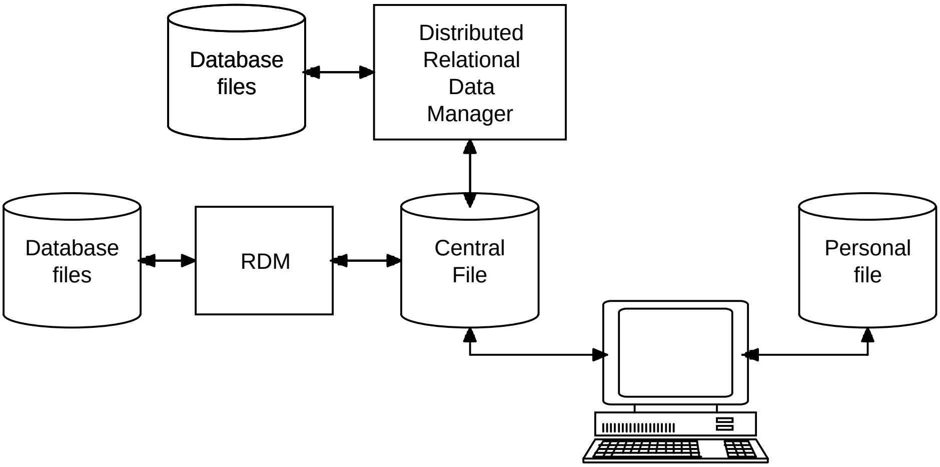
When accessing table data, SUPRA Server controls database navigation, security, and integrity. SUPRA Server rejects any updates that are not allowed, and SPECTRA returns an explanation to the user. SPECTRA also ensures that any data requirements are met by verifying the data requests you enter before processing them.

SPECTRA accesses SUPRA Server tables and views as its central files. SPECTRA can also access VSAM files in IBM environments and RMS files in VMS environments as a source of central files.

With SPECTRA, you can create personal files containing data copied from central files or entered from other sources. Your personal files are not available to other users unless you share the files.

You use SPECTRA commands to create a set of instructions (a process) to perform tasks. You can write processes to request information, perform calculations, enter data, create reports, update personal files, and so on. You can store processes for repeated use. You can also copy and modify an existing process to create a new process for similar tasks.

You can use SPECTRA in batch mode. Any command available with online SPECTRA is available in batch. You can perform all SPECTRA functions, such as creating and running processes, listing files, issuing HELP commands, and creating and printing personal files. You can also submit a process to run in batch mode while using online SPECTRA. You can write and test your processes online and then use the SUBMIT control command to submit the process to run in batch mode. The following figure shows the relationship between central files, personal files, and processes:



Selecting an option from the Master menu

You begin a SPECTRA session by selecting one of the following options from the Master menu:

- **Central files.** SPECTRA lists any central files you are authorized to access. You may select one of the listed files and use SPECTRA commands to:
  - Browse and edit the central file (edit is only available for RDM views).
  - Copy central file data to a personal file.
- **Personal files.** SPECTRA lists any files you have created. You can use SPECTRA commands to:
  - Create, delete, rename, or copy a file.
  - Change the file layout and field descriptions.
  - Browse and edit file contents.
  - Move or copy data between files.
  - Add or delete fields in a file.
- **Processes.** SPECTRA lists your defined processes. You can use SPECTRA commands to:
  - Create, delete, change, or rename a process.
  - Select an existing process to perform a task using process commands.
- **User's guide.** SPECTRA displays the table of contents for the online user's guide.

Splitting screens for dual use

You can divide a SPECTRA screen into two sections (or windows). Each window has its own command line, and you may perform two operations simultaneously. For example, you can edit using one window and display a file process or the Help menu in the other window.

After you unsplit the screen, you can switch between windows by entering EXCHANGE. You may then perform operations on the displayed screen without affecting the other window—both windows remain available.

Using SPECTRA commands

SPECTRA provides two types of easy-to-use commands to browse data and to define your reports:

- **Control commands**. Use to perform immediate actions at the terminal:
  - Browse and select central and personal file data
  - Copy central file data into a personal file
  - Dynamically update personal file data
  - Print files, processes, and help text
  - Copy, delete, and rename personal files and processes
  - Split the screen and display two windows
  - Exchange one screen for another
  - List names of central and/or personal files or processes
  - Print selected data
  - Sort personal files
  - Back up and restore personal files and processes
- **Process commands**. Use to create a set of instructions that perform tasks such as:
  - Creating reports
  - Entering data
  - Updating personal file or RDM data
  - Performing calculations
  - Retrieving information

Using the specialized facilities of SPECTRA

SPECTRA offers the following facilities to help you manipulate your data and provide meaningful reports:

- Built-in functions
- Arithmetic operators
- Statistics
- Comparison symbols and operators
- Display control
- Null support

Customizing reports with built-in functions

SPECTRA has several built-in functions that you can use in a process to customize titles and footers and to control paging. The following functions are available:

- **DATE.** Prints the current date.
- **TIME.** Prints the current time.
- **WEEKDAY.** Prints the current day of the week.
- **LINECOUNT.** Counts the number of lines.
- **PAGECOUNT.** Counts the number of pages.

Calculating statistics

SPECTRA provides the following statistical functions:

- **ANY.** Returns YES if any of a set of records meets a stated condition; returns NO if all records fail to meet the condition.
- **AVERAGE.** Computes the average value of a series of numeric values.
- **COUNT.** Counts the number of times a specific value occurs.
- **EVERY.** Returns YES if every record in a set of records meets a stated condition; returns NO if any records fail to meet the condition.
- **MAX.** Prints the maximum value for a field in a set of records.
- **MIN.** Prints the minimum value for a field in a set of records.
- **TOTAL.** Adds a series of values you select in a set of records.

Comparing values

SPECTRA allows you to select data using one or more of the following comparison symbols or operators:

Symbol	Definition
=	Selects a value that is equal to the specified value.
<	Selects the values less than the specified amount.
>	Selects the values greater than the specified amount.
<>	Selects all values other than the specified value.
NOT=	Selects values not equal to the specified value.
<=	Selects values less than or equal to the specified value.
NOT>	Selects values not greater than the specified values.
>=	Selects values greater than or equal to the specified value.
NOT<	Selects values not less than the specified value.
AND	Selects values based on two different comparisons which must both be true.
BETWEEN	Selects values that fall between two specified values.
CONTAINS	Searches a textual value for the specified pattern.
ENDS	Searches for fields ending with a specified value.
MAYBE	Selects values that are null or that fall within the selection criteria.
OR	Selects values based on two different comparisons. If either comparison is true, the value is selected.
SAME	Selects rows with the same values from two or more files.
STARTS	Searches for fields beginning with a specified value.

Controlling the display

You can use the following special characters in a process to control the displayed data:

Character	Definition
&	Joins (concatenates/connects) text values together.
TRIM	Removes blanks from text values.
@	Extracts parts (substrings) of textual information.
:	Controls the output display. You can specify the column width and/or supply an edit mask for the column.

Using null support

SPECTRA supports null values in both personal files and central files. SPECTRA displays null values as blanks. SPECTRA tracks whether a field value is missing and whether it is blank or zero.

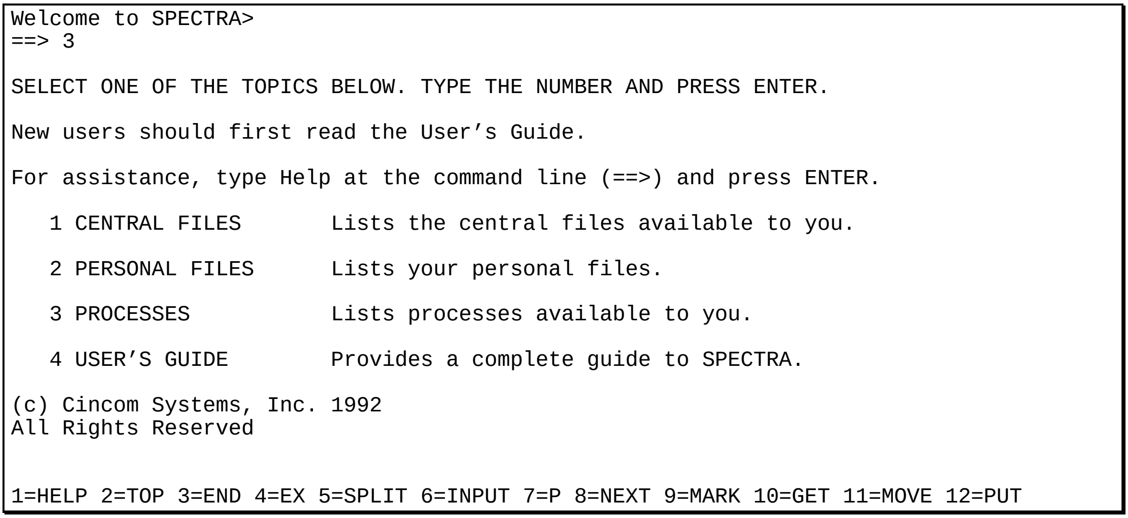
You can use the NULL keyword as follows:

- In Edit and Browse mode to select database on a null value.
- In processes to retrieve data from central and personal files based on whether a field contains a null value. You can also update, delete, and insert null values into SPECTRA personal files.

Running a SPECTRA process

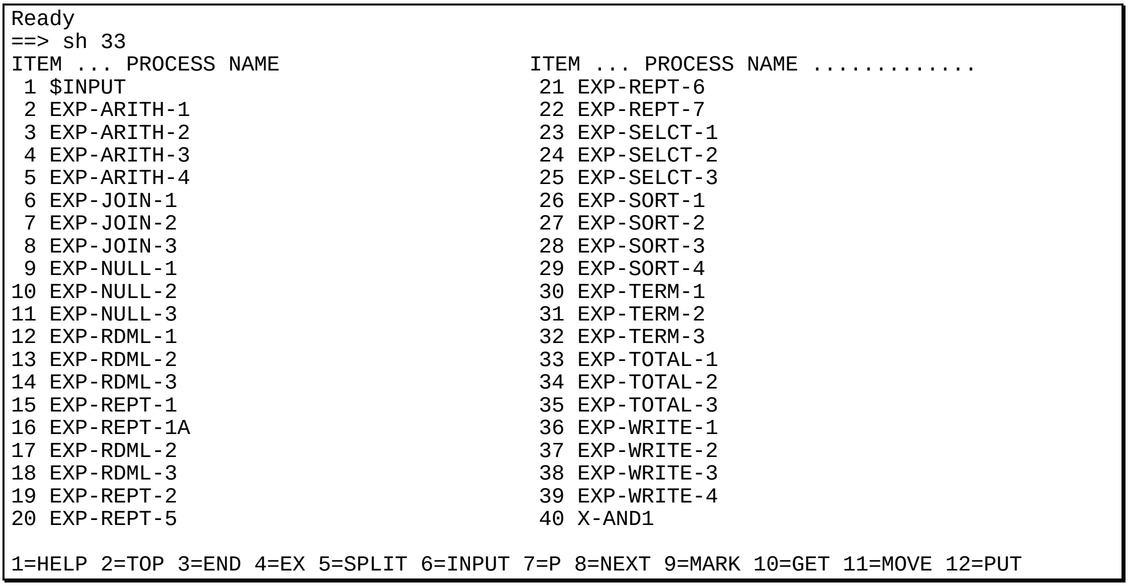
The following screens show how to run an existing SPECTRA process. The screens show a process that creates a report on each manifest number. After signing on to SPECTRA, you select the function you want to perform from the Master menu. SPECTRA prompts you through all subsequent steps.

The following screen shows the Master menu. You enter a 3 to view a list of processes.



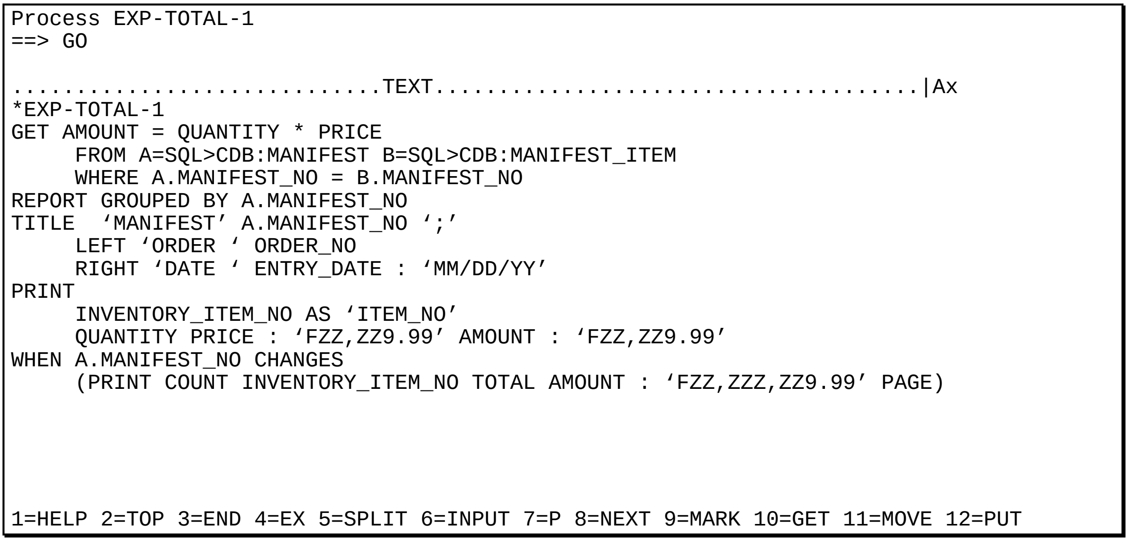
```
Welcome to SPECTRA>
==> 3

SELECT ONE OF THE TOPICS BELOW. TYPE THE NUMBER AND PRESS ENTER.

New users should first read the User’s Guide.

For assistance, type Help at the command line (==>) and press ENTER.

   1 CENTRAL FILES       Lists the central files available to you.

   2 PERSONAL FILES      Lists your personal files.

   3 PROCESSES           Lists processes available to you.

   4 USER’S GUIDE        Provides a complete guide to SPECTRA.

(c) Cincom Systems, Inc. 1992
All Rights Reserved


1=HELP 2=TOP 3=END 4=EX 5=SPLIT 6=INPUT 7=P 8=NEXT 9=MARK 10=GET 11=MOVE 12=PUT
```

On the screen that lists the available processes, you can enter an abbreviated version of the SHOW command to display the process EXP-TOTAL-1.



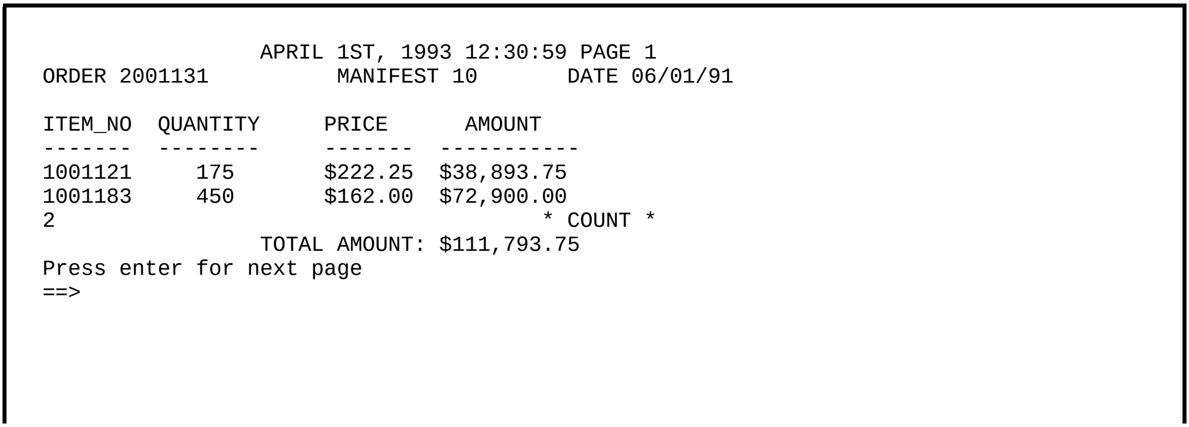
```
Ready
==> sh 33
ITEM ... PROCESS NAME                   ITEM ... PROCESS NAME .............
 1 $INPUT                                21 EXP-REPT-6
 2 EXP-ARITH-1                           22 EXP-REPT-7
 3 EXP-ARITH-2                           23 EXP-SELCT-1
 4 EXP-ARITH-3                           24 EXP-SELCT-2
 5 EXP-ARITH-4                           25 EXP-SELCT-3
 6 EXP-JOIN-1                            26 EXP-SORT-1
 7 EXP-JOIN-2                            27 EXP-SORT-2
 8 EXP-JOIN-3                            28 EXP-SORT-3
 9 EXP-NULL-1                            29 EXP-SORT-4
10 EXP-NULL-2                            30 EXP-TERM-1
11 EXP-NULL-3                            31 EXP-TERM-2
12 EXP-RDML-1                            32 EXP-TERM-3
13 EXP-RDML-2                            33 EXP-TOTAL-1
14 EXP-RDML-3                            34 EXP-TOTAL-2
15 EXP-REPT-1                            35 EXP-TOTAL-3
16 EXP-REPT-1A                           36 EXP-WRITE-1
17 EXP-RDML-2                            37 EXP-WRITE-2
18 EXP-RDML-3                            38 EXP-WRITE-3
19 EXP-REPT-2                            39 EXP-WRITE-4
20 EXP-REPT-5                            40 X-AND1

1=HELP 2=TOP 3=END 4=EX 5=SPLIT 6=INPUT 7=P 8=NEXT 9=MARK 10=GET 11=MOVE 12=PUT
```

SPECTRA displays the process. Notice that this process performs a join, selecting data from two SQL tables. You enter GO to execute the process.



```
Process EXP-TOTAL-1
==> GO

.............................TEXT......................................|Ax
*EXP-TOTAL-1
GET AMOUNT = QUANTITY * PRICE
     FROM A=SQL>CDB:MANIFEST B=SQL>CDB:MANIFEST_ITEM
     WHERE A.MANIFEST_NO = B.MANIFEST_NO
REPORT GROUPED BY A.MANIFEST_NO
TITLE  ‘MANIFEST’ A.MANIFEST_NO ‘;’
     LEFT ‘ORDER ‘ ORDER_NO
     RIGHT ‘DATE ‘ ENTRY_DATE : ‘MM/DD/YY’
PRINT
     INVENTORY_ITEM_NO AS ‘ITEM_NO’
     QUANTITY PRICE : ‘FZZ,ZZ9.99’ AMOUNT : ‘FZZ,ZZ9.99’
WHEN A.MANIFEST_NO CHANGES
     (PRINT COUNT INVENTORY_ITEM_NO TOTAL AMOUNT : ‘FZZ,ZZZ,ZZ9.99’ PAGE)





1=HELP 2=TOP 3=END 4=EX 5=SPLIT 6=INPUT 7=P 8=NEXT 9=MARK 10=GET 11=MOVE 12=PUT
```

SPECTRA displays the resulting report. Notice that it provides a count of the item numbers and a total of the amount.



```
                 APRIL 1ST, 1993 12:30:59 PAGE 1
ORDER 2001131          MANIFEST 10       DATE 06/01/91

ITEM_NO  QUANTITY     PRICE      AMOUNT
-------  --------     -------  -----------
1001121     175       $222.25  $38,893.75
1001183     450       $162.00  $72,900.00
2                                      * COUNT *
                 TOTAL AMOUNT: $111,793.75
Press enter for next page
==>
```

Managing SPECTRA administrative functions

The DBA is responsible for handling SPECTRA administrative functions. These functions include:

- Accessing external files
- Maintaining user profiles
- Maintaining the Personal File System
- Recovering from failure
- Producing User Profile reports

Accessing external files

The SPECTRA INPUT, OUTPUT, IMPORT, EXPORT, IMPORTALL, and EXPORTALL commands allow SPECTRA to access external files.

The INPUT and OUTPUT commands allow you to use an external file within a process for reporting. You use the INPUT command to read data from an external file into a SPECTRA process. You use the OUTPUT command to write data from a SPECTRA process to an external file. INPUT and OUTPUT describe the file layout including the length of each field.

The EXPORT, IMPORT, EXPORTALL, and IMPORTALL commands allow you to copy personal files or processes to or from an external file. You use these commands to back up your data or to share data with other SPECTRA users.

Maintaining user profiles

SPECTRA user profiles provide an additional layer of security for SUPRA Server users. To use SPECTRA with SUPRA, you must be defined both as a SUPRA Server user and as a SPECTRA user. The SPECTRA user definition is maintained in the user profile.

The DBA defines the SPECTRA user profiles, which control feature availability for each user. SPECTRA stores user profiles on the USER_PROFILES personal file, which can only be accessed by the DBA.

Each user profile provides flags which the DBA activates or deactivates. A user profile also provides the user name, language, date and time default formats, a default printer, several user-defined fields, and statistics and accounting information.

The user flags specify whether a user can:

- Read central files during EDIT
- Read central files from a process
- Change central file contents during EDIT
- Change central file contents with a process
- Create personal files
- Create a process
- Execute a process
- Execute a process online
- Show the contents of a process
- Access external files
- Import files

The statistics and accounting information for a user includes the following:

- Maximum number of calls allowed to central, personal, external, or sort files for batch processing. If the maximum limit is reached, SPECTRA stops executing.
- Maximum number of seconds SPECTRA should run a request (moving data or running a process) before SPECTRA interrupts.
- Maximum number of calls allowed to central, personal, external, or sort files for online processing. If the limit is reached, SPECTRA interrupts.
- How many records can be processed after the initial online interrupt message before another message is issued.
- How much disk space is available for personal files and processes.
- How many blocks can be processed for sort files before a process is canceled.
- Current count of personal file blocks used.
- Number of records processed from central files, personal files, external files, and sort files.
- Total amount of time, in seconds, the user has been signed on to SPECTRA.

Maintaining the personal file system

SPECTRA personal files exist in a single RMS file referred to as the PFS file. The contents of each block in the PFS file are managed by the SPECTRA personal file system. The DBA is responsible for maintaining this hierarchy of files. Refer to the *SPECTRA Administration Guide*, P26-9220, for additional information on maintaining this file system.

Recovering from a failure

SPECTRA manages a buffer pool that is written to the PFS file at various commit points, and SPECTRA guarantees the PFS file to be intact to the most recent commit point. Refer to the *SPECTRA Administration Guide*, P26-9220, for additional information on using the PFS file and other means for recovery.

Producing user profile reports

SPECTRA provides four processes you can use to produce reports about user profiles. You can use these processes as they are, or you can copy and modify them to use as a base for your own reporting. The four reports are:

- **Users.** Reports on all of the SPECTRA users.
- **Statistics.** Reports on the user statistics.
- **Blocks.** Reports on the percentage of the allocated blocks a user has used.
- **Authorization.** Reports on each user’s authorization.

A

SUPRA Server documentation synopses

The following synopses provide general information about the SUPRA Server PDM manual series. Some of the SUPRA Server tools are optional; therefore, you may not have all the manuals listed. The documents are listed by publication number within the series.

PDM/RDM documentation series (VMS)

The following are synopses for the manuals included in the PDM/RDM documentation series for VMS environments:

P25-0022 ***SUPRA Server PDM Messages and Codes Reference Manual (PDM/RDM Support for UNIX & VMS)***

Audience General (users of SUPRA Server 2.*x* (VMS), 1.*x* (UNIX) series)

Synopsis Explains error messages and codes generated by SUPRA Server 1.*x* (UNIX) and its associated facilities.

P25-0130 ***SUPRA Server PDM System Administration Guide (VMS)***

Audience VMS system administrators, database administrators, and systems programmers

Synopsis Describes SUPRA Server components, including the PDM, Directory, and RDM, and how they fulfill data processing needs. Provides set-up information such as the logical names needed to run SUPRA Server and the Directory structure SUPRA Server requires. Explains procedures for initiating SUPRA Server components and guidelines for maintaining optimum performance of SUPRA Server. Sections include:

- Implementing the required logical names for SUPRA Server
- Setting up the Directory
- Establishing the required command files to initiate SUPRA Server components
- Controlling PDM usage with PDM operator commands
- Obtaining optimum performance of SUPRA Server
- Writing user exits

P25-0240 *SUPRA Server PDM Programming Guide (UNIX & VMS)*

Audience Application programmers

Synopsis Provides the information programmers need to access and manipulate the database.

For VMS users, this includes:

- Understanding the RDM and Relational Data Manipulation Language (RDML)
- Using RDML commands for COBOL, FORTRAN, and BASIC programs
- Using the DBAID utility subset
- Understanding the PDM and Physical Data Manipulation Language (PDML)
- Using PDML commands

For UNIX users, this includes:

- Understanding the PDM and Physical Data Manipulation Language (PDML)
- Using PDML commands
- Using the Programmer’s Report

P25-1120 *SUPRA Server PDM Directory Views (VMS)*

Audience

Database administrators familiar with SUPRA Server and having some knowledge of DBAID, MANTIS, or SPECTRA. (SPECTRA is not available in the OpenVMS Alpha environment.)

Synopsis

This manual gives descriptions of Cincom-supplied views of the Directory database. These views describe the entities held on the Directory, providing read-only access to all user database definitions. Sections include:

- Instructions for setting up global view files and the logical names you need to access the Directory views
- Instructions for using the Entity views
- Instructions for using the Relationship views
- Instructions for using the special views, such as those which perform a reverse linkpath scan, or the view that displays view access definitions
- Descriptions of the syntax of the programmer’s reports provided to illustrate the data type of the Directory View columns
- Programmer’s reports for the Comment views

P25-2260 *SUPRA Server PDM Database Administration Guide (UNIX & VMS)*

Audience Database administrators

Synopsis This manual provides procedures for designing and maintaining a database description using a menu-driven tool called SUPRA DBA. Sections include:

- Normalizing data
- Creating and maintaining a database description
- Defining and running recovery methods available through SUPRA DBA
- Validating, compiling, printing and formatting a database
- Creating and maintaining views (VMS only)
- Performing administrative tasks including defining user names and changing the structure of live databases

P25-6220 *SUPRA Server PDM Utilities Reference Manual (UNIX & VMS)*

Audience Database administrators

Synopsis Step-by-step instructions for using the DBA, Fast, Database Transfer, and Database Verify Utilities, including instructions for calculating disk space requirements.

P25-8220 *SUPRA Server PDM RDM Administration Guide (VMS)*

Audience Database administrators operating in OpenVMS environments

Synopsis This manual contains a description of RDM and general guidelines on manipulating data with RDM, as well as practical application of RDM. Sections include:

- Creating views
- Changing data with RDM
- Defining base views
- Maintaining referential integrity with RDM
- Defining derived views
- Copying with physical and logical database changes
- Creating global and bound views
- Using DBAID to define and test views
- RDM reports and how to produce them
- Sample RDM reports
- Sample validation user exits

P26-0675 *SUPRA Server Glossary*

Audience General

Synopsis Alphabetically presents SUPRA Server terms and their definitions.

P26-9062 *SUPRA Server Digest*

Audience Database administrators, application programmers, MANTIS and SPECTRA users

Synopsis This manual provides an in-depth technical overview of SUPRA Server. Sections include:

- How SUPRA Server insulates users from the physical data structure
- An introduction to the components that comprise SUPRA Server: the SUPRA Server Directory, PDM, RDM, SPECTRA, and MANTIS
- Maximum and minimum values for entities defined on a SUPRA Server database
- Generating RDM reports and using RDML
- How SUPRA Server PDM works; the physical structure of SUPRA Server databases and the recovery techniques provided
- Synopses of the manuals in the SUPRA Server documentation series

T25-2263 *SUPRA Server PDM VMS Tutorial*

Audience Database administrators, programmers, other users new to SUPRA Server

Synopsis This manual contains detailed instructions for creating a database description in UNIX environments. Sections include:

- ♦ Creating a database description
- ♦ Validating and compiling a database
- ♦ Formatting a database

Index